[1], Peter Wainwright[1]

[a] *Center for Population Biology, University of California, Davis, United States*

% rfishbase: programmatic access for exploring, manipulating and visualizing FishBase data from R % Carl Boettiger; Peter Wainwright % March 28, 2012

Center for Population Biology, University of California, Davis, United States

Abstract

We introduce a package that provides interactive and programmatic access to the FishBase repository. This package allows us to interact with data on over 30,000 fish species in the rich statistical computing environment, `R`. We illustrate how this direct, scriptable interface to FishBase data enables better discovery and integration essential for large-scale comparative analyses. We provide several examples to illustrate how the package works, and how it can be integrated into such as phylogenetics packages `ape` and `geiger`.

keywords

R | vignette | fishbase

Introduction

```
Error: there is no package called 'rfishbase'
```

```
Error: there is no package called 'xtable'
```

FishBase (fishbase.org) is an award-winning online database of information about the morphology, trophic ecology, physiology, ecotoxicology, reproduction, economic relevance of the world's fish, organized by species (Froese and Pauly 2012). FishBase was developed in collaboration with the United Nations Food and Agriculture Organization and is supported by a consortium of nine research institutions. In addition to its web-based interface, FishBase provides machine readable XML files for

```
Error in sprintf("%d", length(fish.data)) : object 'fish.data' not found
```

of its species entries.

To facilitate the extraction, visualization, and integration of this data in research, we have written the `rfishbase` package for the R language for statistical computing and graphics (R Development Core Team 2012). R is a freely available open source computing environment that is used extensively in ecological research, with a large library of packages built explicitly for this purpose (Kneib 2007).

In this paper, we illustrate how the `rfishbase` package is synchronized with the FishBase database, describe its functions for extracting, manipulating and visualizing data, and then illustrate how these functions can be combined for more complicated analyses. Lastly we illustrate how having access to FishBase data through R allows a user to interface with other kinds of analyses, such as comparative phylogenetics software.

[*] Corresponding author.

A programmatic interface

The `rfishbase` package works by creating a cached copy of all data on FishBase currently available in XML format. Caching increases the speed of queries and places minimal demands on the FishBase server, which in its present form is not built to support direct access to application programming interfaces (APIs). A cached copy is included in the package and can be loaded in to R using the command:

```
data(fishbase)
```

This loads a copy of all available data from FishBase into the R list, `fish.data`, which can be passed to the various functions of `rfishbase` to extraction, manipulation and visualization. The online repository is frequently updated as new information is uploaded. To get the most recent copy of FishBase, update the cache instead. The update may take up to 24 hours. This copy is stored in the working directory with the current date and can be loaded when finished.

```
updateCache()
loadCache("2012-03-26fishdata.Rdat")
```

Loading the database creates an object called fish.data, with one entry per fish species for which data was successfully found, for a total of

```
Error in sprintf("%d", length(fish.data)) : object 'fish.data' not found
```

species.

Not all the data available in FishBase is included in these machine-readable XML files. Consequently, `rfishbase` returns taxonomic information, trophic description, habitat, distribution, size, life-cycle, morphology and diagnostic information. The information returned in each category is provided as plain-text, consequently `rfishbase` must use regular expression matching to identify the occurrence of particular words or patterns in this text corresponding to data of interest.

Quantitative traits such as length, maximum known age, spine and ray counts, and depth information are provided consistently for most species, allowing `rfishbase` to extract this data directly. Other queries require pattern matching. While simple text searches within a given field are usually reliable, the `rfishbase` search functions will take any regular expression query, which permits logical matching, identification of number strings, and much more. The interested user should consult a reference on regular expressions after studying the simple examples provided here to learn more.

Data extraction, analysis, and visualization

The basic function data extraction function in `rfishbase` is `which_fish()`. The function takes a list of fishbase data (usually the entire database, `fish.data`, or a subset thereof, as illustrated later) and returns an array of those species matching the query. This array is given as a list of true/false values for every species in the query. This return structure has several advantages which we illustrate by example.

Here is a query for reef-associated fish (mention of "reef" in the habitat description), and second query for fish that have "nocturnal" in their trophic description:

```
reef <- which_fish("reef", "habitat", fish.data)
```

```
Error: could not find function "which_fish"
```

```
nocturnal <- which_fish("nocturnal", "trophic", fish.data)
```

```
Error: could not find function ”which_fish”
```

One way these returned values are commonly used is to obtain a subset of the database that meets this criteria, which can then be passed on to other functions. For instance, if we want the scientific names of these reef fish, we use the `fish_names` function. Like the `which_fish` function, it takes the database of fish, `fish.data` as input. In this case, we pass it just the subset that are reef affiliated,

```
reef_species <- fish_names(fish.data[reef])
```

```
Error: could not find function ”fish_names”
```

Because our `reef` object is a list of logical values (true/false), we can combine this in intuitive ways with other queries. For instance, give us the names for all fish that are nocturnal and not reef associated,

```
nocturnal_nonreef_orders <- fish_names(fish.data[nocturnal & !reef],
    ”Class”)
```

```
Error: could not find function ”fish_names”
```

Note that this time we have also specified that we want taxanomic Class of the fish matching the query, rather than the species names. `fish_names` will allow us to specify any taxanomic level for it to return.

Quantitative trait queries work like `fish_names`, taking a the FishBase data and returning the requested information. For instance, the function `getSize` returns the length (default), weight, or age of the fish in the query:

```
age <- getSize(fish.data, ”age”)
```

```
Error: could not find function ”getSize”
```

`rfishbase` can also extract a table of quantitative traits from the morphology field, describing the number of vertebrate, dorsal and anal rays and spines,

```
morphology_numbers <- getQuantTraits(fish.data)
```

```
Error: could not find function ”getQuantTraits”
```

and extract the depth range (extremes and usual range) from the habitat field,

```
depths <- getDepth(fish.data)
```

```
Error: could not find function ”getDepth”
```

The real power of programatic access the ease with which we can combine, visualize, and statistically test a custom compilation of this data. To do so it is useful to organize a collection of queries into a data frame. Here we combine the queries we have made above and a few additional queries into a data frame in which each row represents a species and each column represents a variable.

```
marine <- which_fish(”marine”, ”habitat”, fish.data)
```

```
Error: could not find function ”which_fish”
```

```r
africa <- which_fish(”Africa:”, ”distribution”, fish.data)
```

```
Error: could not find function ”which_fish”
```

```r
length <- getSize(fish.data, ”length”)
```

```
Error: could not find function ”getSize”
```

```r
order <- fish_names(fish.data, ”Order”)
```

```
Error: could not find function ”fish_names”
```

```r
dat <- data.frame(reef, nocturnal, age, marine, africa, length, order)
```

```
Error: object ’reef’ not found
```

This data frame contains categorical data (*e.g.* is the fish a carnivore) and continuous data (*e.g.* weight or age of fish). We can take advantage of the rich data visualization in R to begin exploring this data. These examples are simply meant to be illustrative of the kinds of analysis possible and how they would be constructed.

Fraction of marine species found in the 10 largest orders

```r
biggest <- names(head(sort(table(order), decr = T), 10))
```

```
Error: unique() applies only to vectors
```

```r
ggplot(subset(dat, order %in% biggest), aes(order, fill = marine)) +
    geom_bar()
```

```
Error: object ’dat’ not found
```

Is age correlated with size?

```r
ggplot(dat, aes(age, length, color = marine)) + geom_point(position = ”jitter”,
    alpha = 0.8) + scale_y_log10() + scale_x_log10()
```

```
Error: object ’dat’ not found
```

A statistical test of this pattern:

```r
xtable(summary(lm(data = dat, length ~ age)))
```

```
Error: could not find function ”xtable”
```

Are reef species longer lived than non-reef species in the marine environment?

```r
ggplot(subset(dat, marine), aes(reef, log(age))) + geom_boxplot()
```

```
Error: object ’dat’ not found
```

Comparative studies

Many ecological and evolutionary studies rely on comparisons between taxa to pursue questions that cannot be approached experimentally.
Instead, we rely on evolution to have performed the experiment already. For instance, recent studies have attempted to identify whether reef-associated clades experience greater species diversification rates than non-reef-associated groups (*e.g.* Alfaro et al. 2009). We can easily identify and compare the numbers of reef associated species in different families using the `rfishbase` functions presented above.

In this example, we answer the simpler question ”Are there more reef-associated species in labrids than in gobies?

Get all species in fishbase from the families “Labridae” (wrasses) or “Scaridae” (parrotfishes):

```
labrid <- which_fish(”(Labridae|Scaridae)”, ”Family”, fish.data)
```

```
Error: could not find function ”which_fish”
```

and get all the species of gobies

```
goby <- which_fish(”Gobiidae”, ”Family”, fish.data)
```

```
Error: could not find function ”which_fish”
```

Identify how many labrids are found on reefs

```
labrid.reef <- which_fish(”reef”, ”habitat”, fish.data[labrid])
```

```
Error: could not find function ”which_fish”
```

```
nlabrids <- sum(labrid.reef)
```

```
Error: object ’labrid.reef’ not found
```

and how many gobies are found on reefs:

```
ngobies <- sum(which_fish(”reef”, ”habitat”, fish.data[goby]))
```

```
Error: could not find function ”which_fish”
```

Note that summing the list of true/false values returned gives the total number of matches.
This tells us that there are

```
Error in unique(c(”AsIs”, oldClass(x))) : object ’nlabrids’ not found
```

labrid species associated with reefs, and

```
Error in unique(c(”AsIs”, oldClass(x))) : object ’ngobies’ not found
```

goby species associated with reefs.

Integration of analyses

One of the greatest advantages about accessing FishBase directly through R is the ability to take advantage of the suite of specialized analyses available through R packages. Likewise, users familiar with these packages can more easily take advantage of the data available on FishBase. We illustrate this with an example that combines phylogenetic methods available in R with quantitative trait data available from `rfishbase`.

This series of commands illustrates testing for a phylogenetically corrected correlation between the maximum observed size of a species and the maximum observed depth at which it is found.

load a phylogenetic tree of labrid fish (provided in the package), and some the phylogenetics package `geiger` (L. Harmon et al. 2009).

```
data(labridtree)
library(geiger)
```

Find the species represented on this tree in FishBase

```
myfish <- findSpecies(tree$tip.label, fish.data)
```

```
Error: could not find function "findSpecies"
```

Get the maximum depth of each species and sizes of each species:

```
depths <- getDepth(fish.data[myfish])[, "deep"]
```

```
Error: could not find function "getDepth"
```

```
size <- getSize(fish.data[myfish], "length")
```

```
Error: could not find function "getSize"
```

Drop tips from the phylogeny for unmatched species.

```
data <- na.omit(data.frame(size, depths))
```

```
Error: object 'size' not found
```

```
pruned <- treedata(tree, data)
```

```
Error: object 'tree' not found
```

Use phylogenetically independent contrasts (Felsenstein 1985) to determine if depth correlates with size after correcting for phylogeny:

```
x <- pic(pruned$data[["size"]], pruned$phy)
```

```
Error: object 'pruned' not found
```

```
y <- pic(pruned$data[["depths"]], pruned$phy)
```

```
Error: object ’pruned’ not found
```

```
xtable(summary(lm(y ~ x - 1)))
```

```
Error: could not find function ”xtable”
```

```
ggplot(data.frame(x = x, y = y), aes(x, y)) + geom_point() + stat_smooth(method = lm)
```

```
Error: object ’x’ not found
```

We can also estimate different evolutionary models for these traits to decide which best describes the data,

```
bm <- fitContinuous(pruned$phy, pruned$data[[”depths”]], model = ”BM”)[[1]]
```

```
Error: object ’pruned’ not found
```

```
ou <- fitContinuous(pruned$phy, pruned$data[[”depths”]], model = ”OU”)[[1]]
```

```
Error: object ’pruned’ not found
```

where the Brownian motion model has an AIC score of

```
Error in unique(c(”AsIs”, oldClass(x))) : object ’bm’ not found
```

while the OU model has a score of

```
Error in unique(c(”AsIs”, oldClass(x))) : object ’ou’ not found
```

, suggesting that

```
Error in which.min(list(BM = bm$aic, OU = ou$aic)) :
  object ’bm’ not found
```

is the better model.

Discussion

With more and more data readily available, informatics is becoming increasingly important in ecology and evolution research (Jones et al. 2006), bringing new opportunities for research (Parr et al. 2011; Michener and Jones 2012) while also raising new challenges (Reichman, Jones, and Schildhauer 2011). It is in this spirit that we introduce the `rfishbase` package to provide programmatic access to the data available on the already widely recognized database, FishBase. We believe that such tools can help take greater advantage of the data available, facilitating deeper and richer analyses than would be feasible under only manual access to the data. We hope the examples in this manuscript serve not only to illustrate how this package works, but to help inspire readers to consider and explore questions that would otherwise be too time consuming or challenging to pursue.

In this paper we have introduced the functions of the `rfishbase` package and described how they can be used to improve the extraction, visualization, and integration of FishBase data in ecological and evolutionary research.

The self-updating study

Because analyses using this data are written in R scripts, it becomes easy to update the results as more data becomes available on FishBase. Programmatic access to data coupled with script-able analyses can help ensure that research is more easily reproduced and also facilitate extending the work in future studies (Peng 2011; Merali 2010). This document is an example of this, using a dynamic documentation interpreter program which runs the code displayed to produce the results shown, eliminating the possibility of faulty code (Xie 2012). As FishBase is updated, we can regenerate these results with less missing data. Readers can find the original document which combines the source-code and text on the project's Github page

Limitations and future directions

FishBase contains much additional data that has not been made accessible in its machine-readable XML format. We are in contact with the database managers and look forward to providing access to additional types of data as they become available. Because most of the data provided in the XML comes as plain text rather that being identified with machine-readable tags, reliability of the results is limited by text matching. Meanwhile, improved text matching queries could provide more reliable and other custom information, such as extracting geographic distribution details as categorical variables or latitude/longitude coordinates. FishBase taxonomy is inconsistent with taxonomy provided elsewhere, and additional package functions could help resolve these differences in assignments.

`rfishbase` has been available to R users through the Comprehensive R Archive Network since October 2011, and has an established user base. The project remains in active development to evolve with the needs of its users. Users can view the most recent changes and file issues with the package on its development website on Github, (https://github.com/ropensci/rfishbase) and developers can submit changes to the code or adapt it into their own software.

Programmers of other R software packages can make use of the `rfishbase` package to make this data available to their functions, further increasing the use and impact of FishBase. For instance, the project OpenFisheries makes use of the `rfishbase` package to provide information about commercially relevant species.

Acknowledgements

CB is supported by a Computational Sciences Graduate Fellowship from the Department of Energy under grant number DE-FG02-97ER25308.

References

Alfaro, Michael E., Francesco Santini, Chad D. Brock, Hugo Alamillo, Alex Dornburg, Daniel L. Rabosky, Giorgio Carnevale, and Luke J. Harmon. 2009. "Nine exceptional radiations plus high turnover explain species diversity in jawed vertebrates." *Proceedings of the National Academy of Sciences* 106 (aug): 13410–14. doi:10.1073/pnas.0811087106. http://www.pubmedcentral.nih.gov/artic

Felsenstein, Joseph. 1985. "Phylogenies and the Comparative Method." *The American Naturalist* 125 (jan): 1–15. doi:10.1086/284325. http://www.journals.uchicago.edu/doi/abs/10.1086/284325.

Froese, R., and Daniel Pauly. 2012. "FishBase." World Wide Web electronic publication.. www.fishbase.org.

Harmon, Luke, Jason Weir, Chad Brock, Rich Glor, Wendell Challenger, and Gene Hunt. 2009. "geiger: Analysis of evolutionary diversification." http://cran.r-project.org/package=geiger.

Jones, Matthew B., Mark P. Schildhauer, O. J. Reichman, and Shawn Bowers. 2006. "The New Bioinformatics: Integrating Ecological Data from the Gene to the Biosphere." *Annual Review of Ecology, Evolution, and Systematics* 37 (dec): 519–544. doi:10.1146/annurev.ecolsys.37.091305.110031. http://arjournals.annualreviews.org/doi/abs/10.1146/annurev.ecolsys.37.091305.11003

Kneib, Thomas. 2007. "Introduction to the Special Volume on 'Ecology and Ecological Modelling in R'." *Journal of Statistical Software* 22: 1–7. http://www.jstatsoft.org/v22/i01/paper.

Merali, Zeeya. 2010. "Why Scientific programming does not compute." *Nature*: 6–8.

Michener, William K., and Matthew B. Jones. 2012. "Ecoinformatics: supporting ecology as a data-intensive science." *Trends in Ecology & Evolution* 27 (jan): 85–93. doi:10.1016/j.tree.2011.11.016. http://linkinghub.elsevier.com/retrieve/pii/S016953471100339

Parr, Cynthia S., Robert Guralnick, Nico Cellinese, and Roderic D. M. Page. 2011. "Evolutionary informatics: unifying knowledge about the diversity of life." *Trends in ecology & evolution* 27 (dec): 94–103. doi:10.1016/j.tree.2011.11.001. http://www.ncbi.nlm.nih.gov/pubmed/22154516.

Peng, R. D. 2011. "Reproducible Research in Computational Science." *Science* 334 (dec): 1226–1227. doi:10.1126/science.1213847. http://www.sciencemag.org/cgi/doi/10.1126/science.1213847.

R Development Core Team, The. 2012. "R: A language and environment for statistical computing." Vienna, Austria: R Foundation for Statistical Computing. http://www.r-project.org/.

Reichman, O. J., Matthew B. Jones, and Mark P. Schildhauer. 2011. "Challenges and Opportunities of Open Data in Ecology." *Science* 331 (feb): 703–705. doi:10.1126/science.1197962. http://www.sciencemag.org/cgi/doi/10.1126/science.1197962 http://www.ncbi.nlm.nih.gov/pubmed/21311007.

Xie, Yihui. 2012. "knitr: A general-purpose package for dynamic report generation in R." http://yihui.name/knitr/.